Article

Motion planning and control of redundant manipulators for dynamical obstacle avoidance

Giacomo Palmieri [1]*, Cecilia Scoccia [1], Matteo-Claudio Palpacelli [1] and Massimo Callegari [1]

[1] Department of Industrial Engineering and Mathematical Sciences, Polytechnic University of Marche, Ancona, Italy
* Correspondence: g.palmieri@univpm.it

Abstract: This paper presents a framework for the motion planning and control of redundant manipulators with the added task of collision avoidance. The algorithms that were previously studied and tested by the authors for planar cases are here extended to full mobility redundant manipulators operating in a three-dimensional workspace. The control strategy consists of a combination of off-line path planning algorithms with on-line motion control. The path planning algorithm is used to generate trajectories able to avoid fixed obstacles, detected before the robot starts to move; it is based on the potential fields method combined with a smoothing interpolation that exploits Bézier curves. The on-line motion control is designed to compensate for the motion of the obstacles and to avoid collisions along the kinematic chain of the manipulator; it is realized by means of a velocity control law based on the null space method for redundancy control. A term of the control law takes into account the speed of the obstacles as well as their position. In order to test the algorithms, a set of simulations are presented: the robot KUKA LBR iiwa is controlled in different cases, where fixed or dynamic obstacles interfere with its motion. Simulations are also used to estimate the required computational effort in order to verify the transferability to a real system.

Keywords: Collision avoidance; redundant manipulators; human-robot collaboration

1. Introduction

Robots are nowadays more and more asked to work in unstructured environments. In many cases they are supported by sensor-based safety systems, avoiding fences that typically isolate the cell workspace within the workshop. Moreover, the trend toward collaborative robotics portends to a workshop layout where robots and humans share the workspace and collaborate in many operations, in a dynamical and unforeseeable scenario. In addition to dedicated hardware and design principles, collaborative robotics implies specific control strategies to ensure safety [1,2]. In this sense, collision avoidance control techniques represent a powerful means of improving the safety and flexibility of robots. A number of papers are available in the literature on path planning and obstacle avoidance for mobile robots [3], which is a very common problem. More complex is the case of manipulators, that suffer from the limitations of the workspace and problems of singularity. However, redundant manipulators offer greater dexterity than traditional manipulators, which aids in the development of task-oriented control strategies taking advantage of the additional degrees of freedom. Thus, redundant manipulators are the best candidates for high dexterity tasks with collision avoidance capability [4,5]. Moreover, redundancy can be exploited also with standard manipulators if some of the degrees of freedom of the end-effector, e.g. the orientation angles, can be kept free during motion. Thus, an overall control strategy for a manipulator working in a dynamic environment can be conceived as the combination of:

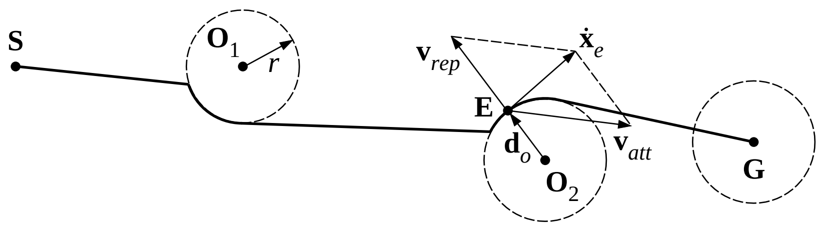


Figure 1. *Potential fields for trajectory planning.*

- An *off-line path planning algorithm*, which plans the trajectory of the robot's end-effector taking into account the possible presence of disturbing obstacles, modifying the path based on the positions of the obstacles before the motion starts;
- An *on-line motion control algorithm*, which controls in real-time the robot compensating for obstacles that are moving, or new obstacles entering the workspace;
- A *redundancy control strategy* that exploits the dexterity of the manipulator to avoid collisions between obstacles and the kinematic chain of the manipulator;
- A robust technique for the *avoidance of singular configurations* during motion.

Dealing with motion planning for obstacle avoidance, several methods are available in the literature [6–10]. A very common approach is to define artificial potential fields, which drive the robot to the target inside the workspace [11–13]. The result of the potential fields is a set of forces, attractive toward the goal and repulsive from the obstacle regions. Typically such forces are associated to velocities applied to the end-effector of the manipulator; then, the trajectory can be obtained by numerical integration.
A further problem in optimal path planning for automation and robotics is the generation of smooth trajectories: an optimal algorithm for trajectory generation must guarantee smoothness in terms of position and velocity in order to be implemented in the controller of a real system. The use of smooth curves, e.g. the Bézier curves [14–19], can help to solve this problem, as already proposed by the authors in [20].
The same principle of repulsive velocities generated by obstacles can be used in order to avoid collisions between obstacles and control points along the kinematic chain of the manipulator: in addition to the motion imposed to end-effector, a repulsive velocity vector can be applied to the point of the robot that is closer to one of the obstacles, adding a task to the control system [21–24]. This kind of approach was studied by the authors in [20] for a planar case; in addition to the standard method a modified repulsive velocity was introduced, improving the capability of the algorithm to compensate for dynamic obstacles.
Inspired by the background described above and making use of previous research conducted by the authors for the planar case, this study presents an extension of the proposed algorithms to a three-dimensional workspace and redundant manipulators with full mobility. The rest of the paper is organized as follows: the algorithms for off-line path planning and on-line motion control are described in *Section 2* and *Section 3* respectively; results obtained by a series of simulated tests are described in *Section 4*; a discussion on the computational effort required and on the issues related to transferability of the control law to a real system is presented in *Section 5*, whereas final conclusions are drawn in *Section 6*.

2. Off-line path planning

The trajectory planning algorithm used to define the motion $\mathbf{x}_e(t)$ of the end-effector, proposed by the authors in [20,25], is extended in the present paper to the 3D case. The generated path allows to reach the target position and to avoid the obstacles that are inside the workspace before the motion of the robot starts. The algorithm is based on the definition of potential fields that generate repulsive and attractive velocity components, defined $\mathbf{v}_{rep}$ and $\mathbf{v}_{att}$ respectively, which drive the end-effector **E**

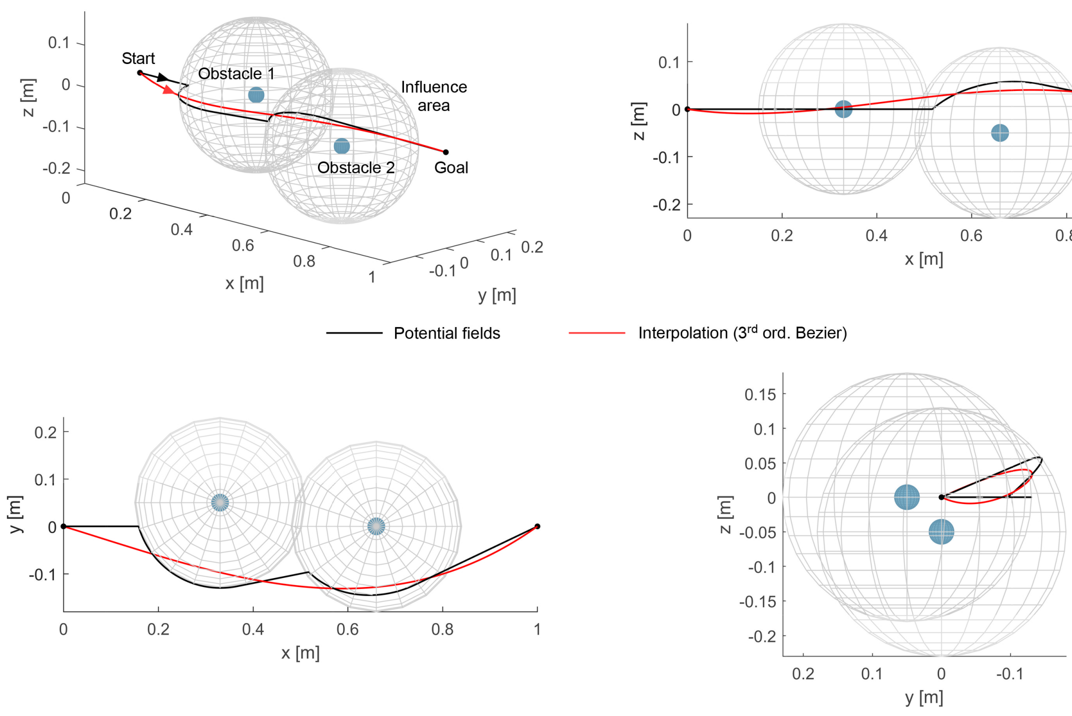


Figure 2. *Example of path planning in 3D space with two obstacles.*

following the minimum potential path towards the goal. As shown in Figure 1, **S** is the initial position of the end-effector, **G** is the goal and $\mathbf{O}_i$ are the obstacles, with their region of influence outlined by their radius r. Moreover: $\mathbf{d}_O = \mathbf{E} - \mathbf{O}_i$ and $\mathbf{d}_G = \mathbf{G} - \mathbf{E}$. Equation (1) defines attractive and repulsive velocities as:

$$\mathbf{v}_{att} = \begin{cases} v_{att}\dfrac{\mathbf{d}_G}{r} & d_G < r \\ v_{att}\hat{\mathbf{d}}_G & d_G \geq r \end{cases} \qquad \mathbf{v}_{rep} = \begin{cases} \dfrac{v_{rep}}{d_O}\left(\dfrac{1}{d_O} - \dfrac{1}{r}\right)\boldsymbol{\nabla} d_O & d_O < r \\ \mathbf{0} & d_O \geq r \end{cases} \tag{1}$$

The end-effector trajectory can be found by numerical integration of the resulting velocity:

$$\dot{\mathbf{x}}_e = \mathbf{v}_{rep} + \mathbf{v}_{att} \qquad \mathbf{x}_e(t + \mathrm{d}t) = \mathbf{x}_e(t) + \dot{\mathbf{x}}_e(t)\mathrm{d}t \tag{2}$$

Then, an interpolation with a fifth order polynomial law is used to generate a timed motion law. Nevertheless, the trajectory resulting from equations (1) and (2) is typically characterized by short-radius curves and sharp corners that may originate high accelerations and vibration problems, as shown in the example of Figure 2: here two obstacles are interposed between the start and goal points preventing the motion over the ideal linear trajectory; the planning algorithm generates the black curve that remains outside the region of influence of the obstacles in all its points, but presents fast changes of directions in two points, i.e. where the trajectory meets the influence spheres of the obstacles. An interpolation procedure that exploits a smoother type of curve, e.g a Bézier curve, can be used to solve the problem. More in detail, given $n+1$ points $\mathbf{P}_0, \mathbf{P}_1, \ldots, \mathbf{P}_n$, where n is the power of the Bezeir curve, the latter is defined as:

$$\mathbf{B}(s) = \sum_{i=0}^{n} \binom{n}{i} \mathbf{P}_i (1-s)^{n-i} s^i, \quad s \in [0,1] \tag{3}$$

Without any loss of generality, let's consider a third order Bézier curve:

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 2 February 2021 doi:10.20944/preprints202102.0096.v1

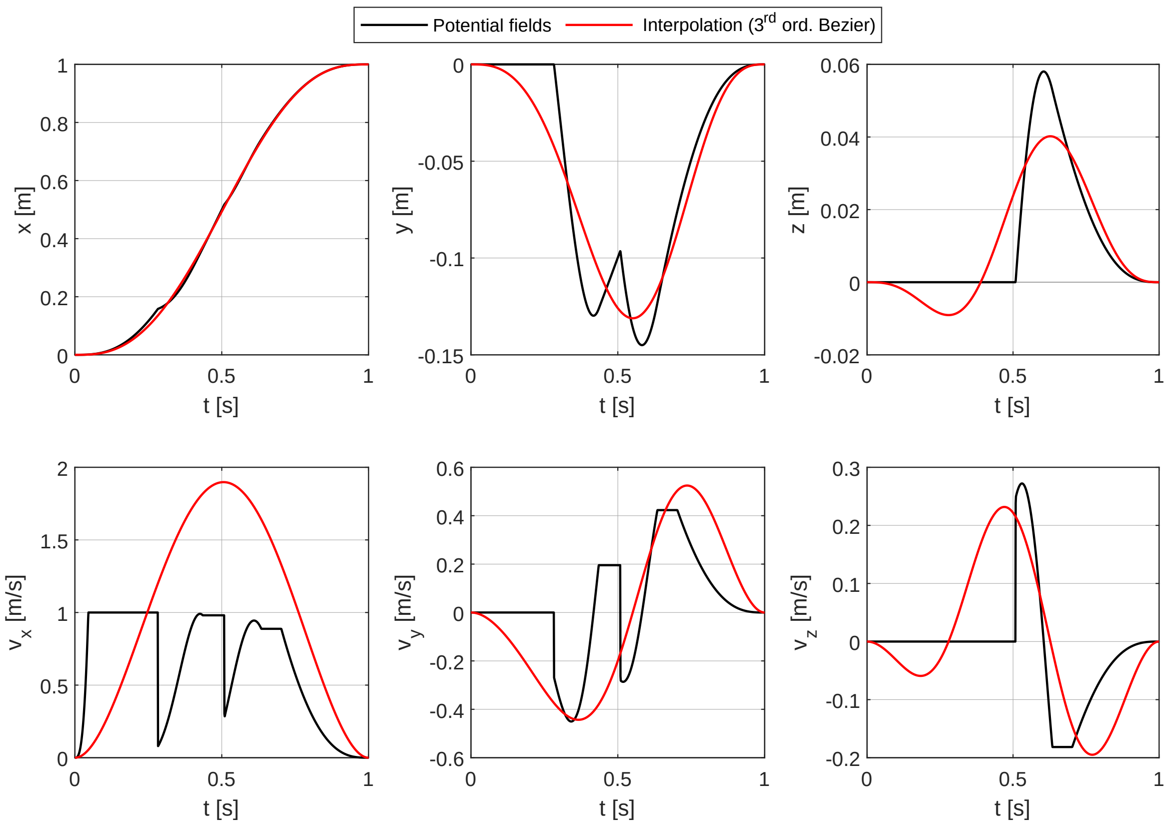


Figure 3. *Comparison between position and velocity profiles of potential fields trajectory and smoothed trajectory related to the example of Figure 2.*

$$\mathbf{B} = \mathbf{P}_0(1-s)^3 + 3\mathbf{P}_1 s(1-s)^2 + 3\mathbf{P}_2 s^2(1-s) + \mathbf{P}_3 s^3 \tag{4}$$

Being points $\mathbf{P}_0$ and $\mathbf{P}_3$ coincident with the starting and goal points respectively, the fitting procedure seeks for points $\mathbf{P}_1$ and $\mathbf{P}_2$ generating the curve $\mathbf{B}$ that best approximates the original one with a least-square metric. A closed-form solution to the problem can be found manipulating equation (4) as follows:

$$\mathbf{B}^T = \begin{bmatrix} 1-s^3 & s^3 \end{bmatrix} \begin{bmatrix} \mathbf{P}_0 & \mathbf{P}_3 \end{bmatrix}^T + \begin{bmatrix} 3s(1-s^2) & 3s^2(1-s) \end{bmatrix} \begin{bmatrix} \mathbf{P}_1 & \mathbf{P}_2 \end{bmatrix}^T \tag{5}$$

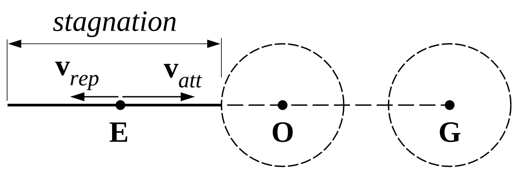
If the curvilinear abscissa s is discretized in m samples, the trajectory $\mathbf{x}_e$ becomes a set of m points. Thus, equation (5) can be written m times for the points $\mathbf{B}_j$, $j = 1 \dots m$, assuming the form:

$$\mathbf{N} = \mathbf{S}_1\mathbf{C}_1 + \mathbf{S}_2\mathbf{C}_2 \tag{6}$$

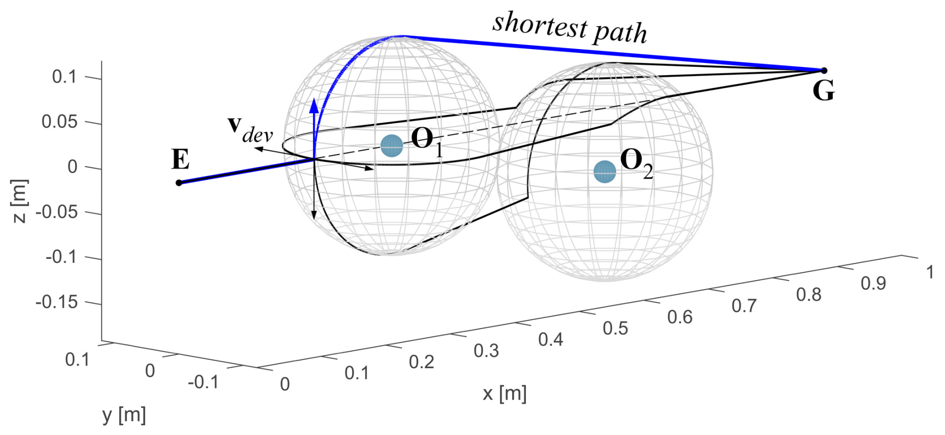
where:

$$\mathbf{N} = \begin{bmatrix} \mathbf{B}_1 \dots \mathbf{B}_m \end{bmatrix}^T \quad \mathbf{S}_1 = \begin{bmatrix} 1-s_1^3 & s_1^3 \\ \vdots & \vdots \\ 1-s_m^3 & s_m^3 \end{bmatrix} \quad \mathbf{S}_2 = \begin{bmatrix} 3s_1(1-s_1^2) & 3s_1^2(1-s_1) \\ \vdots & \vdots \\ 3s_m(1-s_m^2) & 3s_m^2(1-s_m) \end{bmatrix} \tag{7}$$

$$\mathbf{C}_1 = \begin{bmatrix} \mathbf{P}_0 & \mathbf{P}_3 \end{bmatrix}^T \qquad \mathbf{C}_2 = \begin{bmatrix} \mathbf{P}_1 & \mathbf{P}_2 \end{bmatrix}^T \tag{8}$$



a)



b)

Figure 4. *a) Alignment condition between end-effector, obstacle and goal; b) trajectories generated by different directions of* $\mathbf{v}_{dev}$.

A closed-form solution for the optimal set of coefficients $\mathbf{C}_2$ can be easily found manipulating equation (6) and substituting the matrix $\mathbf{N}$ with the analogous matrix $\mathbf{X}$ that is built with the points belonging to the trajectory $\mathbf{x}_e$ found with the potential field algorithm:

$$\mathbf{C}_2 = \mathbf{S}_2{}^{\dagger}(\mathbf{X} - \mathbf{S}_1\mathbf{C}_1) \qquad \mathbf{X} = \begin{bmatrix} \mathbf{x}_{e,1} & \dots & \mathbf{x}_{e,m} \end{bmatrix}^T \tag{9}$$

where † represents the pseudoinverse operator according to the Moore-Penrose definition. The result of the smoothing procedure is shown in Figure 2, where the best-fit 3rd order Bézier curve is plotted in red. A deeper comparison is given by Figure 3, where the Cartesian components of position and velocity of the end-effector are plotted versus time: the path directly obtained by the potential fields algorithm (black plot) presents cusps and discontinuities in the position and velocity profiles respectively, whereas the trajectory related to the Bézier curve (red plot) is smooth both in position and velocity, thus is feasible to be assigned to a real manipulator. Bézier curves of higher order have been also tested, but they suffer from too sharp curvatures and high computational times, thus they are not suitable for the purpose.

When the potential fields approach is used in path planning a particular condition must be taken into account: as shown in Figure 4a, when an obstacle lies exactly on the segment joining the end-effector to the goal point, the attractive and repulsive velocities act in contrast to each other and the end-effector rebounds from the obstacle along such segment, without accomplishing the task. The proposed algorithm overcomes the problem as follows: when the aligning condition between **E**, **O** and **G** is verified, a small deviatoric component $\mathbf{v}_{dev}$ is added to the attractive velocity $\mathbf{v}_{att}$; among infinite directions that can be assigned to $\mathbf{v}_{dev}$ orthogonal to $\mathbf{v}_{att}$, a subset of four of them is selected (aligned with Cartesian axes, with positive or negative directions), whereas its magnitude is constant and predefined. The effect of the deviatoric velocity is to bring the end-effector out of the stagnation line. Obviously, for each one of the selected directions the resulting path is different. Thus, the shortest one is considered for further steps. As an example, Figure 4b shows the trajectories obtained with each one

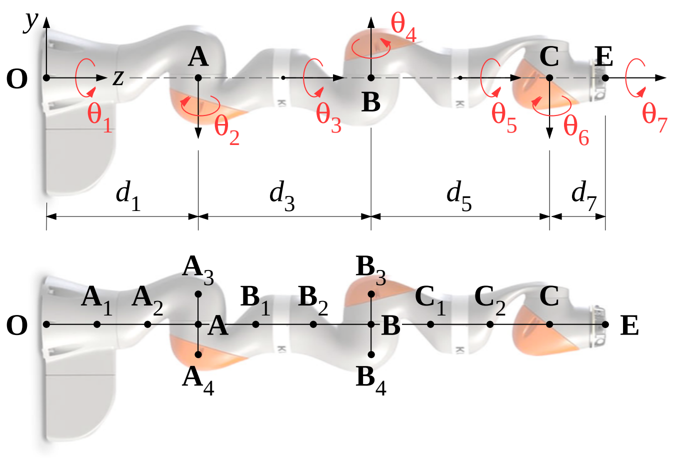


Figure 5. *Kinematic chain (top) and control points (bottom) for the KUKA LBR iiwa robot*

of the four different directions assigned to $\mathbf{v}_{dev}$; among them the shorted one (blue curve) is chosen.
All the issues discussed above in this section concerned the planning of the Cartesian position of the end-effector of the manipulator, that is, the generation of the motion laws $x(t)$, $y(t)$, $z(t)$. A different strategy must be defined to generate a smooth transition between the initial and final orientation of the end-effector: if the orientation at the target point is different from the initial one, a linear transition with the same timing law used for the position is defined for the three parameters used for the representation, e.g. the Euler angles.

3. On-line motion control

Starting from the results obtained by the authors in [20], the mentioned algorithms are implemented in the present work for a redundant manipulator with full mobility, i.e. the KUKA LBR iiwa 14 R820 robot. The kinematic scheme of the manipulator is shown in Figure 5. The pose of the end-effector in the Cartesian space is defined by the vector $\mathbf{x} = [x\ y\ z\ \alpha\ \beta\ \gamma]^T$, where the last three components are the Euler angles according to the ZYZ convention. The seven rotations related to the revolute joints of the serial kinematic chain form the joint space position vector, defined as $\mathbf{q} = [\theta_1\ \theta_2\ \theta_3\ \theta_4\ \theta_5\ \theta_6\ \theta_7]^T$. Thus, the kinematic chain of the manipulator has a redundant degree of freedom with respect to the task. Besides the end-effector **E**, a total of 13 control points (**A**, **B**, **C**, $\mathbf{A_1}$, $\mathbf{A_2}$, $\mathbf{A_3}$, $\mathbf{A_4}$, $\mathbf{B_1}$, $\mathbf{B_2}$, $\mathbf{B_3}$, $\mathbf{B_4}$ $\mathbf{C_1}$, $\mathbf{C_2}$) belonging to the kinematic chain characterizes the manipulator, as shown in the bottom picture of Figure 5.
The forward kinematics of the manipulator can be described by:

$$\mathbf{x} = \mathbf{f}(\mathbf{q}) \tag{10}$$

$$\dot{\mathbf{x}} = \begin{bmatrix} \dot{\mathbf{x}}_p \\ \boldsymbol{\omega} \end{bmatrix} = \begin{bmatrix} \mathbf{J}_p \\ \mathbf{J}_o \end{bmatrix} \dot{\mathbf{q}} = \mathbf{J}(\mathbf{q})\dot{\mathbf{q}} \tag{11}$$

In the previous equations **f** represents the expression of the position forward kinematics and **J** is the (6×7) analytic Jacobian composed by the position and orientation Jacobian matrices $\mathbf{J}_p$ and $\mathbf{J}_o$, each one of dimensions (3×7); the velocity vector $\dot{\mathbf{x}}$ is composed by the position vector $\dot{\mathbf{x}_p}$ and the angular velocity $\boldsymbol{\omega}$, whose expressions are given by:

$$\begin{gathered} \mathbf{x}_p = [\dot{x} \quad \dot{y} \quad \dot{z}]^T \\ w = \left[\dot{\gamma}\cos\alpha\sin\beta - \dot{\beta}\sin\alpha \quad \dot{\beta}\cos\alpha + \dot{\gamma}\sin\alpha\sin\beta \quad \dot{\alpha} + \dot{\gamma}\cos\beta\right]^T \end{gathered} \tag{12}$$

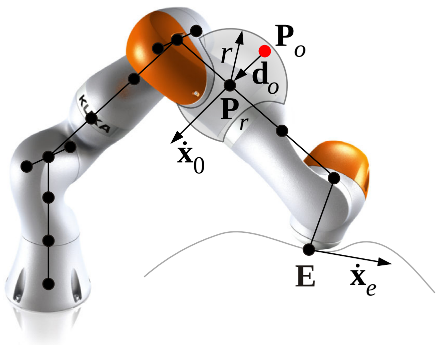


Figure 6. *Velocity of the end-effector* **E** *and of the control point* $\mathbf{P}_r$ *closest to the obstacle* $\mathbf{P}_o$

Due to redundancy, the inverse position kinematics problem is not uniquely defined, but it can be workedout by the following differential formulation:

$$\dot{\mathbf{q}} = \mathbf{J}^{\dagger}\dot{\mathbf{x}} + \mathbf{N}\dot{\mathbf{q}}_0 \tag{13}$$

$$\mathbf{q}(t + \mathrm{d}t) = \mathbf{q}(t) + \dot{\mathbf{q}}(t)\mathrm{d}t \tag{14}$$

The terms of equation (13) are defined as follows: $\dot{\mathbf{q}}_0$ is the joint null space velocity, whose effect is to generate internal motions leaving the pose of end-effector unchanged; $\mathbf{J}^{\dagger} = \mathbf{J}^{\mathbf{T}}(\mathbf{J}\mathbf{J}^{\mathbf{T}})^{-1}$ is the pseudoinverse of the Jacobian matrix $\mathbf{J}$; $\mathbf{N} = \mathbf{I} - \mathbf{J}^{\dagger}\mathbf{J}$ is the projection into the null space of $\mathbf{J}$.
The inverse kinematic problem formulated in equation (13) suffers from numerical problems due to the inversion of the Jacobian matrix when the manipulator is near to a singular configuration, i.e. when the singular values of $\mathbf{J}$ tend to zero. In order to avoid singularity problems, the use of the well known damped least-square method was applied [26]. Such method consists in the substitution of the pseudoinverse of the Jacobian $\mathbf{J}^{\dagger}$ by:

$$\mathbf{J}^{*} = \mathbf{J}^{\mathbf{T}}(\mathbf{J}\mathbf{J}^{\mathbf{T}} + \lambda^2\mathbf{I})^{-1} \tag{15}$$

where λ represents a damping factor that confers a better numerical conditioning to the inversion problem. The value of λ should be a compromise between accuracy (low value) and numerical robustness (high value). An efficient way to determine λ is to define it as a function of the smallest singular value s_{min} of $\mathbf{J}$:

$$\lambda = \begin{cases} 0 & s_{min} \geq \epsilon \\ \left[1 - \left(\frac{s_{min}}{\epsilon}\right)^2\right]\lambda^2_{max} & s_{min} < \epsilon \end{cases} \tag{16}$$

In this way the effect of the approximation vanishes when the smallest singular valuer is greater than a threshold ϵ, where $\lambda \to \lambda_{max}$ when $s_{min} \to 0$, being λ_{max} and ϵ tunable parameters of the algorithm. This kind of approximation introduces a position error that must be subsequently recovered by a proportional term in the control law, as typically done in Closed-Loop Inverse Kinematics (CLIK) control laws [27].
In addition to the basic control law, an obstacle avoidance strategy is introduced: inspired by the null space methods for the control of redundant manipulators, an additional velocity vector is assigned to the control point of the robot which is the closest to one of the obstacles within the workspace, so that the control point can move away form the obstacle, while the motion of the end-effector is not affected. More in detail, referring to the Figure 6, the couple of points $\mathbf{P}_r$ and $\mathbf{P}_o$ at the minimum distance d_o is identified at each time step. The region of influence of each control point is delimited by the radius r. If at a certain time step during the motion the condition $d_o < r$ is verified, a repulsive velocity $\dot{\mathbf{x}}_0$ is

imposed to the relative control point along the direction of $\mathbf{d}_o$. Being the manipulator characterized by one redundant DOF, only one repulsive velocity vector can be assigned at each time step, thus the point to which it is assigned may change over time with the criterion of minimum distance from one of the obstacles. Such point can be also the end-effector: besides the velocity vector $\dot{\mathbf{x}}_e$ assigned by the off-line path planning in order to describe the desired trajectory, the repulsive velocity vector can be applied to $\mathbf{E}$, modifying its trajectory, if the position of an obstacle changes from its initial state or a new obstacle enters the workspace, interfering with the motion of $\mathbf{E}$. In this case, the position of the end-effector drifts from the originally planned trajectory, originating a position error analogous to the one related to the damped least-square approximation for the inversion of the Jacobian matrix. Nevertheless, the CLIK control law can be exploited to recover the error.
In terms of equations, the following expressions (17) and (18) must be imposed in order to assign the two velocity tasks above described:

$$\mathbf{J}\dot{\mathbf{q}} = \dot{\mathbf{x}}_e \tag{17}$$

$$\mathbf{J}_0\dot{\mathbf{q}} = \dot{\mathbf{x}}_e \tag{18}$$

where $\mathbf{J}_0$ represents the Jacobian matrix associated to the velocity of the point $\mathbf{P}_r$.

Thus, equation (13) can be modified in [28,29]:

$$\dot{\mathbf{q}} = \mathbf{J}^*\dot{\mathbf{x}}_c + (\mathbf{J}_0\mathbf{N}^*)(\dot{\mathbf{x}}_0 - \mathbf{J}_0\mathbf{J}^*\dot{\mathbf{x}}_e) \tag{19}$$

where $\mathbf{N}^* = \mathbf{I} - \mathbf{J}^*\mathbf{J}$, $\dot{\mathbf{x}}_c = \dot{\mathbf{x}}_e + \mathbf{K}\mathbf{e}$ is the corrected end-effector velocity, $\mathbf{K}$ a positive-defined gain matrix (for simplicity defined as $\mathbf{K} = k_e\mathbf{I}$) and $\mathbf{e}$ is the position error between the desired position $\mathbf{x}_e$ and the actual position $\mathbf{x}$. The error $\mathbf{e}$ can be represented by [26]:

$$\mathbf{e} = \begin{bmatrix} \mathbf{e}_t \\ \mathbf{e}_o \end{bmatrix} = \begin{bmatrix} \mathbf{p}_e - \mathbf{p} \\ \frac{1}{2}(\mathbf{n} \times \mathbf{n}_e + \mathbf{s} \times \mathbf{s}_e + \mathbf{a} \times \mathbf{a}_e) \end{bmatrix} \tag{20}$$

where the translation error is given by the (3×1) vector $\mathbf{e}_t$ and the orientation error is given by the (3×1) vector $\mathbf{e}_o$. The end-effector position is expressed by the (3×1) position vector $\mathbf{p}$, whereas its orientation by the (3×3) rotation matrix $\mathbf{R} = [\mathbf{n}\ \mathbf{s}\ \mathbf{a}]$, with $\mathbf{n},\ \mathbf{s},\ \mathbf{a}$ being the unit vectors of the end-effector frame.
The first term of equation (19) guarantees the exact velocity of the end effector with minimum joints speed. The second term drives the motion of the point $\mathbf{P}_r$ of the robot, satisfying the collision avoidance additional task. The choice of $\dot{x}_0$ is a critical point of the algorithm. The proposed strategy is to change $\dot{x}_0$ according to the distance from the obstacle. To avoid a discontinuity and give smoothness to the motion, two weighting coefficients, a_h and a_v, are introduced:

$$\dot{\mathbf{x}}_0 = a_v v_{rep}\hat{\dot{\mathbf{x}}}_0 \tag{21}$$

$$\dot{\mathbf{q}} = \mathbf{J}^*\dot{\mathbf{x}}_c + a_h(\mathbf{J}_0\mathbf{N}^*)(a_v v_{rep}\hat{\dot{\mathbf{x}}}_0 - \mathbf{J}_0\mathbf{J}^*\dot{\mathbf{x}}_e) \tag{22}$$

Thus, a nominal repulsive velocity v_{rep} is modulated by a_v as a function of the distance d_o, whereas a_h acts with a weight that balances the effect of the homogeneous solution (related to the collision avoidance) with respect to the total.
More in detail, let r be the control distance that defines the region of influence of a control point, r_{min} the distance under which no action is possible, and r_m an intermediate critical distance between r_{min} and r. No influence of the obstacle is desired if $d_o > r$, whereas the algorithm fails (the robot stops) if $d_o < r_{min}$. Between these limits the weighting coefficients must vary continuously from 0 to 1, as shown in Figure 7.

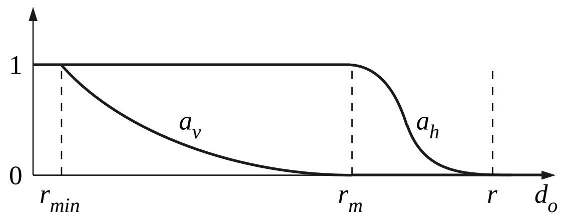


Figure 7. *Weighting coefficients a_v and a_h as functions of the distance d_o*

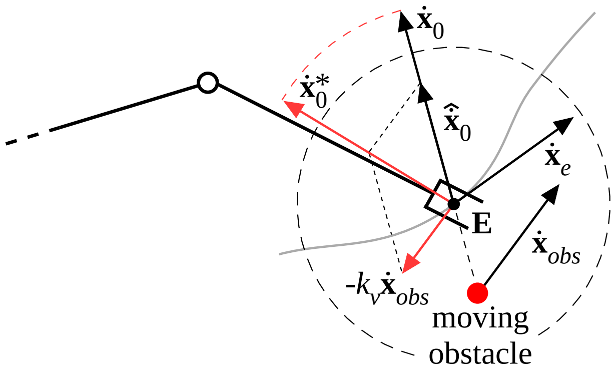


Figure 8. *Modified repulsive velocity.*

$$a_v = \begin{cases} \left(\dfrac{d_o - r_m}{r_{min} - r_m}\right)^2 & d_o < r_m \\ 0 & d_o \geq r_m \end{cases} \tag{23}$$

$$a_h = \begin{cases} 1 & d_o \leq r_m \\ \dfrac{1}{2}\left[1 - \cos\left(\pi\dfrac{d_o - r_m}{r - r_m}\right)\right] & r_m < d_o < r \\ 0 & d_o \geq r \end{cases} \tag{24}$$

In order to improve the ability of the algorithm to avoid moving obstacles, a modification of the control law expressed by equation (22) can be introduced changing the definition of the repulsive velocity [20]: it is reasonable that not only the position, but also the velocity of an obstacle should influence the control of the robot. As an example, if the end-effector moves toward an obstacle having an orthogonal velocity, it is preferable to modify the trajectory of the end-effector pushing it in the direction opposite to the obstacle velocity. Thus, if the end-effector is the control point closest to the obstacle, the repulsive velocity vector $\dot{\mathbf{x}}_0$ is modified as follows:

$$\dot{\mathbf{x}}_0^* = \dot{x}_0 \frac{\hat{x}_0 - k_v \dot{\mathbf{x}}_{obs}}{\sqrt{1 + {k_v}^2 \dot{\mathbf{x}}_{obs}^2}} \tag{25}$$

where, as described in Figure 8, $\dot{\mathbf{x}}_{obs}$ is the obstacle velocity, $\dot{\mathbf{x}}_0$ is the repulsive velocity along the direction of $\mathbf{d}_0$, k_v is a gain term and $\dot{\mathbf{x}}_0^*$ is the modified repulsive velocity applied to the end-effector in combination with the planned velocity $\dot{\mathbf{x}}_e$. As a consequence, the direction of the repulsive velocity is modified, whereas its magnitude doesn't change; furthermore, for $k_v = 0$ the effect vanishes.

4. Results

The algorithms presented in the previous sections were tested by Matlab simulations over a wide range of conditions. In this section three different examples are shown; in all of them a common

Table 1. *Parameters values used for simulations.*

r [m]	r_m [m]	r_{min} [m]	v_{rep} [m/s]	v_{att} [m/s]	T [s]
0.18	0.15	0.12	10	1	2
dt [s]	k_e	k_v	λ_{max}	ϵ	$\dot{\theta}_{max}$ [rad/s]
10^{-3}	100	100	10^{-3}	10^{-3}	π

set of parameters is used (Table 1) and a threshold $\dot{\theta}_{max}$ is imposed for the angular velocity of all joints. Thus, if the control asks for a joint speed higher than $\dot{\theta}_{max}$, the velocity saturates avoiding dangerous situations, whereas the consequent positioning error is recovered by means of the corrective proportional term during the following part of the motion.

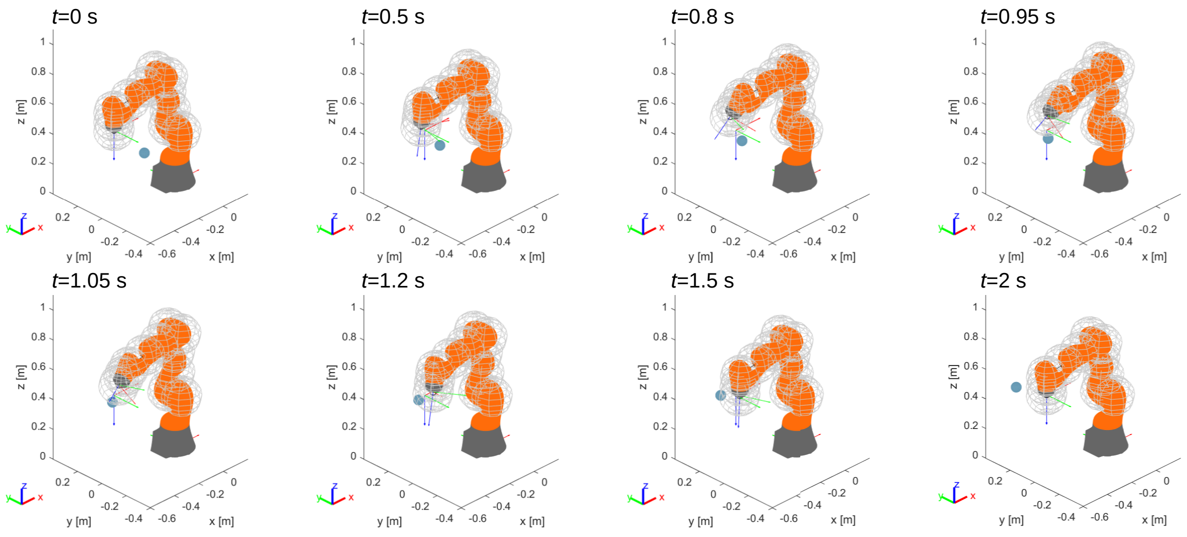


Figure 9. *Example 1: avoidance of a dynamic obstacle interfering with the end-effector in a fixed position.*

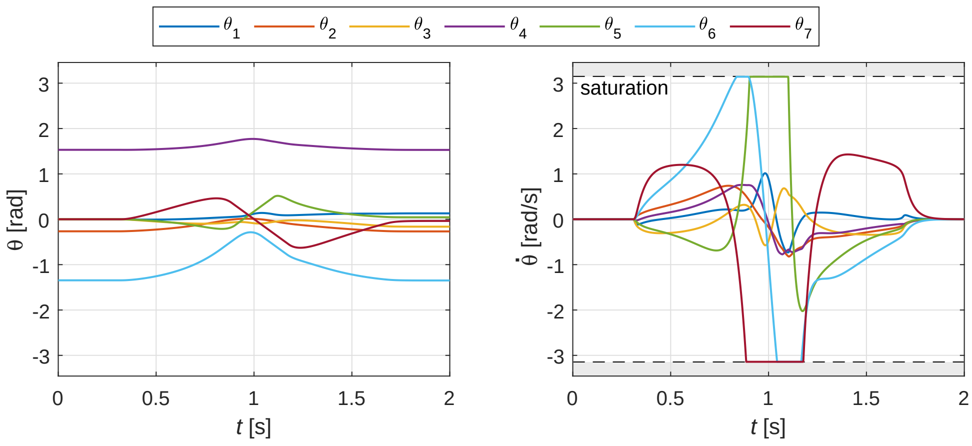


Figure 10. *Example 1: joint rotations and speeds for the motion shown in Figure 9.*

4.1. Example 1

In the first example the robot is fixed while a dynamic obstacle is interfering with the end-effector. The obstacle moves linearly in the horizontal plane along y direction, with a speed of 0.25 m/s. Figure

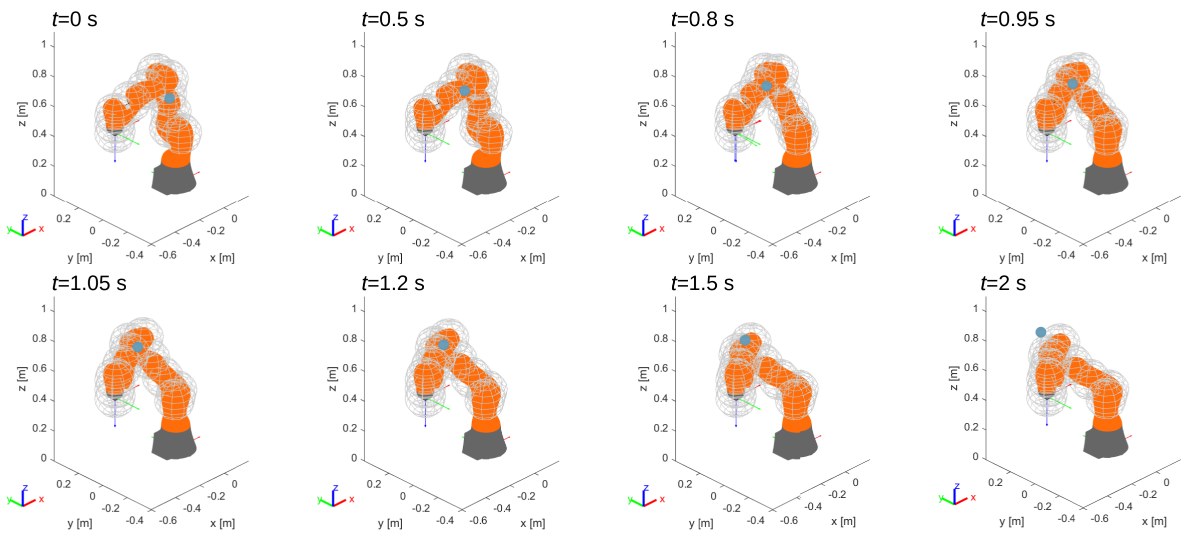


Figure 11. *Example 2: avoidance of a dynamic obstacle interfering with an internal point of the manipulator.*

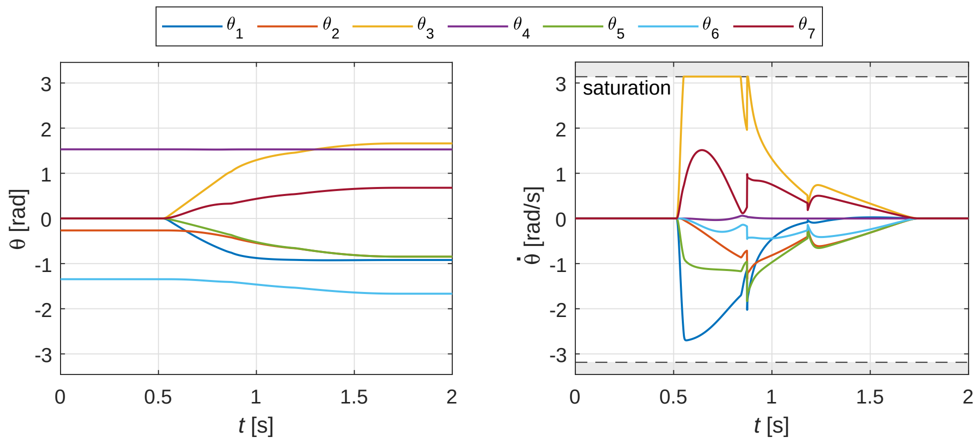


Figure 12. *Example 2: joint rotations and speeds for the motion shown in Figure 11.*

9 shows eight steps of the motion: when the obstacle reaches the region of influence of the end-effector point of the robot the control reacts keeping the obstacle outside from the safety sphere with radius r_{min}. Then, the positioning and orientation error generated by the control is suddenly recovered once the obstacle overcomes the region of influence.
The corresponding profiles of joint rotations and speeds are shown in Figure 10: the effect of the obstacle is visible at $t = 0.3\,\mathrm{s}$, where the joint speed curves suddenly increase their magnitude, reaching in some case the saturation in the middle part of the motion. The effect of the redundancy of the kinematics is clearly visible looking at the final values of joint angles, which are different from the initial ones, despite the Cartesian pose of the manipulator is the same.

4.2. Example 2

In this example the obstacle moves again linearly in the horizontal plane along y direction, with a speed of $0.25\,\mathrm{m/s}$, but, differently from the previous case, the end-effector position is not altered by the obstacle. As shown in Figure 11, the risk of collision occurs in a point of the robot belonging to an intermediate link of the manipulator. Thanks to the redundancy of the kinematic chain, the control is able to reconfigure the manipulator without changing the pose of the end-effector. Due to the saturation of the speed of the third joint (Figure 12), a small motion of the end-effector can be noticed

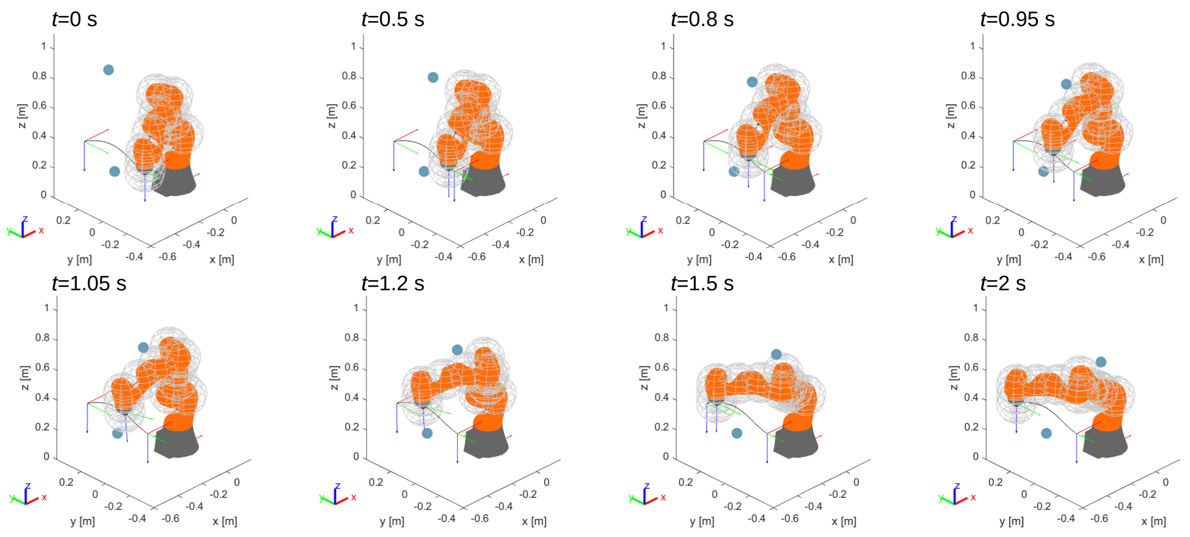


Figure 13. *Example 3: avoidance of two obstacles.*

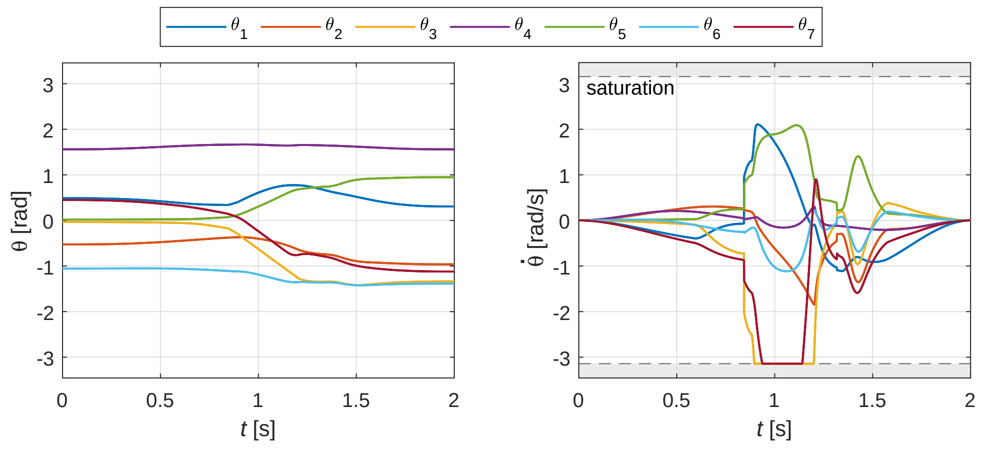


Figure 14. *Example 3: joint rotations and speeds for the motion shown in Figure 13.*

($t = 0.8$ s), which however is quickly recovered by the control. In a similar case where the task could be performed without speed saturation, the absence of motion of the end-effector would be guaranteed.

4.3. Example 3

The third and more complex example presents two obstacles (Figure 13). The first, fixed, is located under the linear path of the robot. Since its proximity to the path is less than the radius r of the end-effector's region of influence, the off-line trajectory planner generates a Bézier curve that avoids the obstacle by passing over it. Once the trajectory is assigned and the motion of the manipulator starts, a second obstacle, dynamic, moves linearly at the speed of 0.25 m/s along a path that intersects the kinematic chain of the robot. The effect of the second obstacle is visible in Figure 14, where a sudden change of the velocity profiles is visible at approximately $t = 0.9$ s. Despite the interference of the two obstacles and the speed saturation for some joint, the control is able to execute the task.

5. Discussion

The simulations presented in the previous section are intended to verify the correctness of the method and to investigate the computational effort needed to implement the related algorithms in a real system. As a matter of fact, the final objective of this research is to transfer the control framework

to a real system, now under construction in laboratory, where the robot will be equipped with a vision system composed by multiple 3D cameras, able to acquire the surrounding environment and extrapolate the position of objects and humans within the workspace. Fixed obstacles simulated in this study can be thought of as fixed objects inadvertently left inside the workspace, such as furniture elements or mechanical tools, whereas dynamic obstacles may represent humans operating in the shared workspace.
What is expected as a critical issue in the implementation on the real system is the speed of execution of the control loop. Based on the results presented in this paper, referring in particular to the most demanding case of example 3, it can be summarized that for the execution of the off-line path planning algorithm the average computational time is approximately 0.15s, whereas the on-line motion control is able to run with a cycle time of approximately $2 \cdot 10^{-3}$ s, comparable with rates typically used in the communication between external controllers and main controllers of robots. Furthermore, a standard laptop (i7 CPU @1.8 GHz, 16 GB RAM) was used for simulations, thus a reduction of computational times can be expected if a more performing hardware is used.

6. Conclusions

In this paper an obstacle avoidance strategy for robots moving in dynamically varying environments is presented and verified by simulation. Two novel contributions to algorithms are introduced: first, the trajectory planned by the potential fields method is smoothed using a best-fit interpolation with Bézier curves; second, a modified repulsive velocity for the end-effector is introduced in order to consider also the velocity of the obstacles, improving the avoidance ability in dynamic environments. The algorithms previously tested for simplified planar cases, are extended to a full mobility redundant manipulator operating in a spatial workspace. In addition to confirming the effectiveness of the control strategy, simulations give an estimation of the computational effort required to execute the algorithms, which is in line with typical requirements of robot controllers and can be improved by using higher performing hardware.

Acknowledgments: This work is partly funded by the project URRA' "usability of robots and reconfigurability of processes: enabling technologies and use cases", on the topics of User-Centered Manufacturing and Industry 4.0, which is part of the project EU ERDF, POR MARCHE Region FESR 2014/2020–AXIS 1–Specific Objective 2–ACTION 2.1,"HD3Flab-Human Digital Flexible Factory of the Future Laboratory", coordinated by the Polytechnic University of Marche..

References

1. Pedrocchi, N.; Vicentini, F.; Matteo, M.; Tosatti, L.M. Safe Human-Robot Cooperation in an Industrial Environment. *International Journal of Advanced Robotic Systems* **2013**, *10*, 27.
2. Vicentini, F. Collaborative Robotics: A Survey. *Journal of Mechanical Design* **2020**, *143*.
3. Tang, S.H.; Kamil, F.; Khaksar, W.; Zulkifli, N.; Ahmad, S.A. Robotic motion planning in unknown dynamic environments: Existing approaches and challenges. 2015 IEEE International Symposium on Robotics and Intelligent Sensors (IRIS), 2015, pp. 288–294.
4. Zlajpah, L.; Nemec, B. Kinematic control algorithms for on-line obstacle avoidance for redundant manipulators. IEEE/RSJ international conference on intelligent robots and systems. IEEE, 2002, Vol. 2, pp. 1898–1903.
5. Bottin, M.; Rosati, G. Trajectory Optimization of a Redundant Serial Robot Using Cartesian via Points and Kinematic Decoupling. *Robotics* **2019**, *8*.
6. Gasparetto, A.; Boscariol, P.; Lanzutti, A.; Vidoni, R. Path planning and trajectory planning algorithms: A general overview. In *Motion and operation planning of robotic systems*; Springer, 2015; pp. 3–27.
7. Han, B.; Luo, X.; Luo, Q.; Zhao, Y.; Lin, B. Research on Obstacle Avoidance Motion Planning Technology of 6-DOF Manipulator. International Conference on Geometry and Graphics. Springer, 2021, pp. 604–614.
8. Willms, A.R.; Yang, S.X. Real-time robot path planning via a distance-propagating dynamic system with obstacle clearance. *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)* **2008**, *38*, 884–893.

9. Xu, P.; Wang, N.; Dai, S.L.; Zuo, L. Motion Planning for Mobile Robot with Modified BIT* and MPC. *Applied Sciences* **2021**, *11*, 426.
10. Wang, J.; Liu, S.; Zhang, B.; Yu, C. Manipulation Planning with Soft Constraints by Randomized Exploration of the Composite Configuration Space. *International Journal of Control, Automation and Systems* **2021**, pp. 1–12.
11. Jung, J.H.; Kim, D.H. Local Path Planning of a Mobile Robot Using a Novel Grid-Based Potential Method. *International Journal of Fuzzy Logic and Intelligent Systems* **2020**, *20*, 26–34.
12. Xu, X.; Hu, Y.; Zhai, J.; Li, L.; Guo, P. A novel non-collision trajectory planning algorithm based on velocity potential field for robotic manipulator. *International Journal of Advanced Robotic Systems* **2018**, *15*, 1729881418787075.
13. Lee, K.; Choi, D.; Kim, D. Potential Fields-Aided Motion Planning for Quadcopters in Three-Dimensional Dynamic Environments. AIAA Scitech 2021 Forum, 2021, p. 1410.
14. Chen, L.; Qin, D.; Xu, X.; Cai, Y.; Xie, J. A path and velocity planning method for lane changing collision avoidance of intelligent vehicle based on cubic 3-D Bezier curve. *Advances in Engineering Software* **2019**, *132*, 65–73.
15. Kawabata, K.; Ma, L.; Xue, J.; Zhu, C.; Zheng, N. A path generation for automated vehicle based on Bezier curve and via-points. *Robotics and Autonomous Systems* **2015**, *74*, 243–252.
16. Corinaldi, D.; Carbonari, L.; Callegari, M. Optimal Motion Planning for Fast Pointing Tasks With Spherical Parallel Manipulators. *IEEE Robotics and Automation Letters* **2018**, *3*, 735–741. doi:10.1109/LRA.2018.2789845.
17. Corinaldi, D.; Callegari, M.; Angeles, J. Singularity-free path-planning of dexterous pointing tasks for a class of spherical parallel mechanisms. *Mechanism and Machine Theory* **2018**, *128*, 47 – 57. doi:https://doi.org/10.1016/j.mechmachtheory.2018.05.006.
18. Hsu, T.W.; Liu, J.S. Design of smooth path based on the conversion between η 3 spline and Bezier curve. 2020 American Control Conference (ACC). IEEE, 2020, pp. 3230–3235.
19. Yu, X.; Zhu, W.; Xu, L. Real-time Motion Planning and Trajectory Tracking in Complex Environments based on Bézier Curves and Nonlinear MPC Controller. 2020 Chinese Control And Decision Conference (CCDC). IEEE, 2020, pp. 1540–1546.
20. Scoccia, C.; Palmieri, G.; Palpacelli, M.C.; Callegari, M. A collision avoidance strategy for redundant manipulators in dynamically variable environments: on-line perturbations of off-line generated trajectories. *Machines* **submitted**.
21. Maciejewski, A.A.; Klein, C.A. Obstacle avoidance for kinematically redundant manipulators in dynamically varying environments. *The international journal of robotics research* **1985**, *4*, 109–117.
22. Zhang, H.; Jin, H.; Liu, Z.; Liu, Y.; Zhu, Y.; Zhao, J. Real-time kinematic control for redundant manipulators in a time-varying environment: Multiple-dynamic obstacle avoidance and fast tracking of a moving object. *IEEE Transactions on Industrial Informatics* **2019**, *16*, 28–41.
23. Abhishek, T.S.; Schilberg, D.; Doss, A.S.A. Obstacle Avoidance Algorithms: A Review. IOP Conference Series: Materials Science and Engineering. IOP Publishing, 2021, Vol. 1012(1), p. 012052.
24. Safeea, M.; Neto, P.; Bearee, R. On-line collision avoidance for collaborative robot manipulators by adjusting off-line generated paths: An industrial use case. *Robotics and Autonomous Systems* **2019**, *119*, 278 – 288.
25. Scoccia, C.; Palmieri, G.; Palpacelli, M.C.; Callegari, M. Real-Time Strategy for Obstacle Avoidance in Redundant Manipulators. The International Conference of IFToMM ITALY. Springer, 2020, pp. 278–285.
26. Chiaverini, S.; Siciliano, B.; Egeland, O. Review of the damped least-squares inverse kinematics with experiments on an industrial robot manipulator. *IEEE Transactions on Control Systems Technology* **1994**, *2*, 123–134.
27. Melchiorre, M.; Scimmi, L.S.; Pastorelli, S.P.; Mauro, S. Collison Avoidance using Point Cloud Data Fusion from Multiple Depth Sensors: A Practical Approach. 2019 23rd International Conference on Mechatronics Technology (ICMT). IEEE, 2019, pp. 1–6.
28. Cefalo, M.; Magrini, E.; Oriolo, G. Sensor-based task-constrained motion planning using model predictive control. *IFAC-PapersOnLine* **2018**, *51*, 220–225.
29. Lee, K.K.; Buss, M. Obstacle avoidance for redundant robots using Jacobian transpose method. 2007 IEEE/RSJ International Conference on Intelligent Robots and Systems. IEEE, 2007, pp. 3509–3514.